Data Structure

Subject: CSW2(CSE3141)

Session: Feb 2024 to April 2024

Branch: CSE&CSIT

Section : All

Q1. Create a class BSTNode that contains a member 'info' to store a number, with 'left' referring to the left child and 'right' referring to the right child. Provide the necessary constructor. Additionally, create a method to insert a node into a binary search tree.

Q2. Write a program to add methods to the binary search tree created in Q1 for traversing the tree in pre-order, in-order, and post-order. Invoke the above methods for execution.

Q3. Create a class Country containing members for name and population, along with a constructor and necessary methods. Additionally, create a class BNode with a member 'info' to store a country object, 'left' to refer to the left child, and 'right' to refer to the right child. Provide the required constructor. Finally, create another class BST with a member 'root', along with the necessary constructor and a method to insert a node into the binary search tree.

Q4. Extend the BST created in Q3 by adding methods to traverse the tree in level order, find the node with the maximum population (find-max), and find the node with the minimum population (find-min). Invoke these methods for execution.

Q5. Construct a binary search tree (BST) from the given array of elements: {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}. Include a method called 'CreateTree' to construct the binary search tree from a sorted array. This method takes an array of integers as input and constructs the tree recursively using a binary search algorithm.

Q6. Determine if a given binary tree is a binary search tree. You will use an 'isBST()' method, which takes the maximum and minimum range of the values of the nodes.

Q7. Remove node x from the binary search tree and reorganize the nodes to maintain its necessary properties.
There are three cases in the deletion process. Let us denote the node that needs to be deleted as x:
Case 1: Node x has no children.
Case 2: Node x has one child.
Case 3: Node x has two children.

Q8. Write a program to implement the graph using adjacency matrix representation and adjacency list representation. Create methods to display the adjacency matrix and adjacency list of the graph.

Q9. Create a class Graph with a linked list member to represent N number of vertices. Implement the required constructor. Add a method to the Graph class for reading a graph and storing it in an adjacency list representation.
Include a depth-first search (DFS) method in the Graph class to traverse the vertices of the graph, and a main method to invoke all the methods.

Q10. Write a Java program to traverse a graph using breadth-first search (BFS) and provide the final output of the code. (Use the ArrayDeque collection.)